DESENVOLVIMENTO E IMPLEMENTAÇÃO DE UMA BASE COMPUTACIONAL ORIENTADA A OBJETOS PARA APLICAÇÕES EM SISTEMAS DE ENERGIA ELÉTRICA

Marcelo N. Agostini*
agostini@labplan.ufsc.br

Ildemar C. Decker*
decker@labplab.ufsc.br

Aguinaldo S. e Silva†
aguinald@labspot.ufsc.br

*LABPLAN, Departamento de Engenharia Elétrica - Universidade Federal de Santa Catarina, 88040-900 Florianópolis, SC, Brasil

†LABSPOT, Departamento de Engenharia Elétrica - Universidade Federal de Santa Catarina, 88040-900 Florianópolis, SC, Brasil

ABSTRACT

This paper describes the application of the Object-Oriented Modeling (OOM) on the modeling of Power Systems. It is intend to develop a computational object-oriented base to represent, in an efficient way, the physical components of a Power System and their methodologies of analysis and synthesis. The generalization level of the classes' structure allows future specializations, on the side of implementing, in an integrated way, a number of methodologies of analysis and synthesis in the area. The motivations for this study are the current limitations imposed by the methods and languages traditionally used for the development of software for Power Systems, for example, the low level of integration between modules and the necessity of high efforts in maintenances and updates. The methodology of the study consists of modeling the Power Systems and their applications, starting from object-oriented project methods, mainly the Object Modeling Technique (OMT). In this paper two modules, which represent methodologies of analysis are implemented, aiming to validate the classes' structure proposed.

KEYWORDS: Object-oriented modeling, power systems modeling.

RESUMO

Este trabalho descreve a aplicação de técnicas de Modelagem Orientada a Objetos (MOO) na modelagem de Sistemas de Energia Elétrica (SEE). Pretende-se desenvolver uma base computacional orientada a objetos que represente de forma eficiente os componentes físicos dos SEE e suas metodologias de análise e síntese. A estrutura de classes possui um grau de generalização tal que permite especializações futuras no sentido de se implementar, de forma integrada, as mais diversas metodologias de análise e síntese na área. As motivações para o estudo são as atuais limitações impostas pelos métodos e linguagens tradicionalmente utilizados no desenvolvimento dos softwares na área de SEE, tais como baixo nível de integração entre módulos e a necessidade de elevados esforços para manutenções e atualizações. A metodologia de trabalho consiste em se modelar os SEE e suas aplicações, tendo por base métodos de projeto orientados a objetos, com especial atenção ao método

Artigo submetido em 11/12/00
1a. Revisão em 22/11/01
Aceito sob recomendação do Ed. Assoc. Prof. José L.R. Pereira

Object Modeling Tecnique (OMT). Neste trabalho são implementados dois módulos representando metodologias de análise, visando a validação das estruturas propostas.

1 INTRODUÇÃO

As transformações ocorridas nos últimos anos na organização dos Sistemas de Energia Elétrica (SEE), com o estabelecimento de um ambiente competitivo, principalmente na geração e na comercialização, têm sinalizado para o aproveitamento máximo das capacidades das instalações. Ao mesmo tempo, novas metodologias e novos equipamentos (FACTS, por exemplo) são constantemente desenvolvidos, e precisam ser incorporados nas ferramentas de apoio ao setor elétrico de forma rápida e eficiente. Assim, um bom suporte computacional ao desenvolvimento dessas ferramentas torna-se imprescindível (Zhou, 1996; Manzoni et al., 1999).

A maioria dos softwares tradicionais do setor são softwares de grande porte, que têm seus desenvolvimentos baseados em métodos procedurais. Isto lhes impõe certas características, tais como dificuldades na integração entre módulos desenvolvidos por diferentes equipes, e necessidade de elevados investimentos para manutenções e atualizações. Estas características são pouco compatíveis com o novo ambiente que se estabelece para o setor (Neyer et al., 1990; Zhou, 1996; Zhu et al., 1997).

Tendo em vista estes aspectos, e também os avanços na área de engenharia de software, alguns trabalhos têm surgido utilizando Modelagem Orientada a Objetos (MOO) no desenvolvimento de softwares para o setor elétrico. A maioria trata de aplicações em fluxo de potência (Neyer et al., 1990; Hakavik et al., 1994; Foley et al., 1995; Zhou, 1996; Fuerte-Esquivel et al., 1998), encontrando-se também aplicações na área de sistemas de distribuição (Zhu et al., 1997), estabilidade transitória (Manzoni et al., 1999), e planejamento (Handschin et al., 1998).

Seguindo esta tendência, este trabalho objetiva o desenvolvimento de uma base computacional orientada a objetos, a qual possibilite a implementação das mais diversas metodologias para análises e sínteses na área de SEE. Para isso, toma-se como base o método Object Modeling Technique (OMT) (Rumbaugh et al., 1994). As estruturas propostas são implementadas em dois módulos representando metodologias de análise, utilizando-se a linguagem de programação $C++$ (Stroustrup, 1997).

O trabalho está organizado da seguinte forma: a seção 2 apresenta uma conceituação básica da POO, visando facilitar a compreensão daqueles profissionais da área de SEE não familiarizados com o tema. A seção 3 apresenta algumas considerações sobre a modelagem de SEE na forma de objetos. A seção 4 mostra as estruturas de classes desenvolvidas para a representação de SEE. A seção 5 apresenta diagramas funcionais e de estado para dois módulos implementados, cada um representando metodologias de análise. Finalizando, a seção 6 apresenta as conclusões do trabalho.

2 MODELAGEM ORIENTADA A OBJETOS

A MOO pode ser definida como uma técnica de projeto de software, na qual se está interessado em primeira instância na clareza e organização do projeto, tendo por base uma representação clara e eficiente do mundo real (domínio da aplicação), de forma a facilitar ao máximo o desenvolvimento e a manutenção do software (Rumbaugh et al., 1994). Prioriza-se na MOO a *estrutura de dados* do sistema. Este enfoque proporciona ao processo de desenvolvimento do projeto uma base estável e permite a utilização de um único conceito em todo o processo: o *objeto*. Objetos devidamente encapsulados, com interfaces públicas que ocultam suas estruturas internas privadas, ficam protegidos de efeitos colaterais em futuras manutenções no software.

Assim, o objeto é a entidade fundamental da MOO, e possui atributos (características próprias) e métodos (formas de manipular seus atributos). Em tempo de projeto, definem-se as *classes*, as quais definem por sua vez o tipo de cada objeto. Cada objeto é dito como sendo uma instância de uma determinada classe. As classes (ou objetos) relacionam-se entre si através das *associações* (ou *ligações*). Da mesma forma que classes e objetos, diz-se que uma ligação é uma instância de uma associação. As classes contém associações entre si, ou seja, determinados tipos de objetos estarão ligados a outros objetos (Rumbaugh et al., 1994; Coleman et al., 1996; Booch, 1994).

Existem alguns tipos especiais de associações, como *agregação* e *herança*. Quando um objeto é formado pela combinação de outros objetos, diz-se que existe aí uma *agregação*. Já o mecanismo de *herança* permite que uma determinada classe, dita *descendente* (*derivada* ou *subclasse*) herde características de uma outra classe, a *base* (ou *superclasse*). Uma outra propriedade na MOO é o *polimorfismo*, o qual permite sobrecarregar determinadas operações e métodos, de forma que uma mesma operação (ou método) realize diferentes procedimentos, quando aplicada em objetos diferentes (Rumbaugh et al., 1994; Coleman et al., 1996; Booch, 1994).

2.1 Técnicas de Modelagem

Nesta seção são apresentados alguns aspectos do método *Object Modeling Technique* (OMT), uma técnica de projeto de sistemas orientados a objetos desenvolvido principalmente por James Rumbaugh (Rumbaugh et al., 1994).

No método OMT representa-se o sistema através de três visões ortogonais, chamadas modelos: o *Modelo de Objetos*, representado graficamente pelos *Diagramas de Classes*; o *Modelo Funcional*, representado pelos *Diagramas de Fluxo de Dados*; e o *Modelo Dinâmico*, representado pelos *Diagramas de Estados*. O Modelo de Objetos representa os aspectos estáticos e estruturais do sistema; representa seus dados, organizados sob a forma de objetos. O Modelo Funcional descreve *o que* o sistema faz e *com quem*, independente de como ou quando; os diagramas de fluxos de dados mostram as dependências entre os valores de saída e entrada em um determinado procedimento. O Modelo Dinâmico descreve os aspectos de um sistema relacionados ao tempo e à seqüência de operações que assinalam modificações nos estados do sistema; os diagramas de estados mostram *quando* ocorre cada evento.

O método define ainda três etapas distintas no processo de desenvolvimento do software: *Análise, Projeto e Implementação*. A Figura 1 mostra estas etapas.

Por suas características de versatilidade e eficiência, o método OMT é tomado como base para o desenvolvimento deste trabalho.

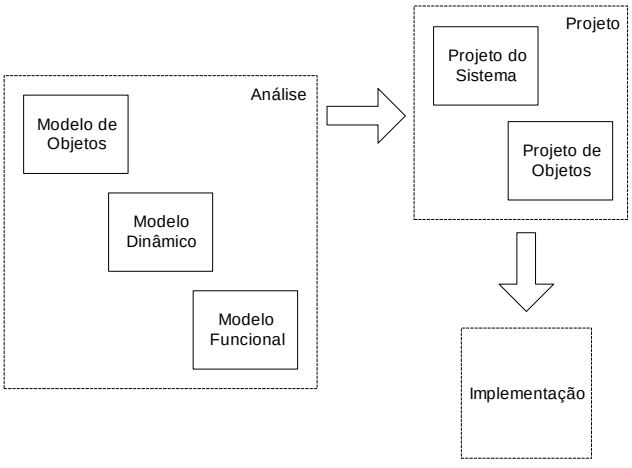

Figura 1: Etapas de Projeto do OMT

3 APLICAÇÃO DE MOO EM SEE

3.1 Adaptabilidade dos SEE à Modelagem Orientada a Objetos

Uma análise prévia da viabilidade da aplicação da MOO no projeto de um determinado sistema computacional é imprescindível para o sucesso do projeto. Fazendo-se esta análise com relação à área de SEE, nota-se que a aplicação da MOO dá-se de forma natural, com benefícios em muitos aspectos. Os SEE têm uma estrutura física bastante adaptável à uma estrutura hierárquica de classes. A forma como os componentes se conectam constituindo a rede elétrica, e também as funcionalidades desses componentes, sugerem o formato geral das estruturas de classes (Zhu et al., 1997; Fuerte-Esquivel et al., 1998; Manzoni et al., 1999).

3.2 Modelagem de Metodologias como Classes

Além das estruturas de classes representativas dos componentes físicos dos SEE, as diversas metodologias de análise e síntese podem ser organizadas em estruturas hierárquicas bem definidas. Rumbaugh et al. (1994) afirma que "*...uma operação que tem características próprias deve ser modelada como classe...*".

Seguindo-se essa idéia, neste trabalho define-se a construção de duas grandes estruturas: uma representativa dos componentes físicos dos SEE, apresentada na seção 4.1, e outra para representar as diversas metodologias de análise e síntese, mostrada na seção 4.2. Recentemente sugerida em alguns trabalhos (Handschin et al., 1998; Fuerte-Esquivel et al., 1998), esta maneira de representar de forma independente os elementos do sistema elétrico (componentes físicos e metodológicos) traz flexibilidade à modelagem. Na construção da estrutura dos componentes físicos dos SEE abstrai-se as características específicas das metodologias a serem implementadas, e vice-versa.

3.3 MOO versus Desempenho

A definição das estruturas de classes é o ponto de partida em um projeto orientado a objetos. Porém essa definição não é uma tarefa fácil (Zhu et al., 1999). Em problemas complexos geralmente existem várias formas possíveis de se representar a hierarquia dos elementos. No caso de SEE, onde os métodos de análise e síntese tendem a serem dispendiosos computacionalmente, uma forma ou outra de representação da estrutura, mesmo estando todas de acordo com a filosofia da MOO, podem implicar

em diferenças significativas de desempenho.

Entende-se que um *bom software orientado a objetos* deve ter, além de características já mencionadas, um bom desempenho computacional. Neyer et al. (1990), trabalho pioneiro na área, relata problemas de desempenho de programas orientados a objetos. Porém, trabalhos mais recentes (Zhou, 1996; Fuerte-Esquivel et al., 1998; Manzoni et al., 1999) têm apresentado melhores resultados para implementações orientadas a objetos, utilizando-se normalmente a linguagem de programação $C++$. Atribui-se essa evolução principalmente ao melhor desempenho dos compiladores mais recentes, os quais incorporam técnicas para a geração de aplicativos bastante eficientes a partir de códigos orientados a objetos (Foley et al., 1995).

Outra questão importante a ser observada nesse aspecto é o constante avanço do hardware. A capacidade de processamento das máquinas disponíveis no mercado aumenta consideravelmente a cada ano. Além dos aspectos citados, os recentes desenvolvimentos e crescente interesse pelo sistema operacional LINUX têm permitido a construção de *clusters de PCs* a baixo custo e com elevada capacidade de processamento.

3.4 Multiplicidade e Expansibilidade das Estruturas de Classes

As estruturas de classes para SEE não devem ser fechadas, e sim prever expansões, mesmo que alterações na estrutura já existente se façam necessárias. Além disso, tais alterações não devem propagar efeitos colaterais pelo restante do código; a estrutura deve estar preparada para alterações dessa natureza, encapsulando seus objetos e definindo adequadamente suas interfaces de comunicação para a troca de mensagens entre os objetos.

Uma analogia pode ser feita, ainda que grosseira, com a arquitetura de um microcomputador PC. As suas primeiras versões (XT) não contemplavam o uso de equipamentos como placas de som, placas de aquisição de imagens, barramento PCI, ou memória DIMM. Posteriormente essas novas tecnologias foram surgindo, e foram sendo adaptadas à estrutura original do PC. O barramento ISA é o mesmo utilizado até hoje em máquinas Pentium. Estruturas internas de funcionamento, tais como o sistema de endereçamento dos dispositivos e o sistema de interrupções para o acionamento desses dispositivos, utilizados nos barramentos ISA, são os mesmos utilizados em novos padrões de barramentos, como o PCI. Placas com as mais diferentes funções são facilmente adaptadas em uma máquina PC, desde que sigam as interfaces definidas previamente (pinagem dos barramentos). Caso surja um dispositivo que não se adapte eficientemente às interfaces existentes, nada impede que se defina uma nova interface (como foi o caso de barramentos como o *Local Bus*, PCI, e mais recentemente o AGP – para placas de vídeo).

4 MODELAGEM DO SISTEMA

4.1 Modelagem dos Componentes Físicos dos SEE

Para a modelagem dos componentes físicos dos SEE é proposta uma estrutura hierárquica, mostrada no diagrama de classes da Figura 2. A estrutura representa os componentes dos SEE como sendo descendentes de três classes abstratas, *Elementos em Derivação* (*ElemDer*), *Elementos Série (ElemSer)* e *Outros Elementos (ElemOutr)*, as quais descendem de uma única classe abstrata *Elemento*. Cada *ElemDer* está conectado a uma barra, enquanto cada *ElemSer* conecta-se a duas barras. Uma barra pode ter conexão com múltiplos componentes série ou em derivação. Derivam de *ElemDer*, classes como *Unidades de Geração (UnidGer), Shunts, Cargas* e *Elementos em Derivação Eletronicamente Controláveis (EDEC)*. As três primeiras são classes concretas, isto é, originam objetos diretamente; a última é abstrata, pois dela derivam modelos. A classe *ElemSer* generaliza os elementos série eletronicamente controláveis (*ESEC*) e as linhas de transmissão ou transformadores (*LT*). Devido às suas semelhanças de modelagem, os componentes linhas de transmissão e transformadores são representados por uma única classe, a qual incorpora características de modelagem comuns a ambos.

Componentes que não se conectam diretamente à barras, mas sim constituem elementos série ou em derivação através de agregados, são generalizados pela classe *ElemOutr*. Desta derivam barras, máquinas síncronas equivalentes (*Maq*), reguladores de tensão (*RAT*), controladores (*Control*), compensadores série controláveis (*CSC*) e compensadores estáticos de reativos (*CER*). A classe *Control* generaliza qualquer tipo de controlador existente nos sistemas elétricos, inclusive Estabilizadores de Sistemas de Potência (ESP ou PSS).

A estrutura de classes da Figura 2 representa ainda o objeto *Sistema de Energia Elétrica* (*SEE*), modelado como um agregado de três ou mais elementos. Considera-se como SEE, um conjunto de pelo menos um gerador alimentando uma carga, ambos conectados através de uma barra.

Na Figura 3 apresenta-se um exemplo de objeto formado

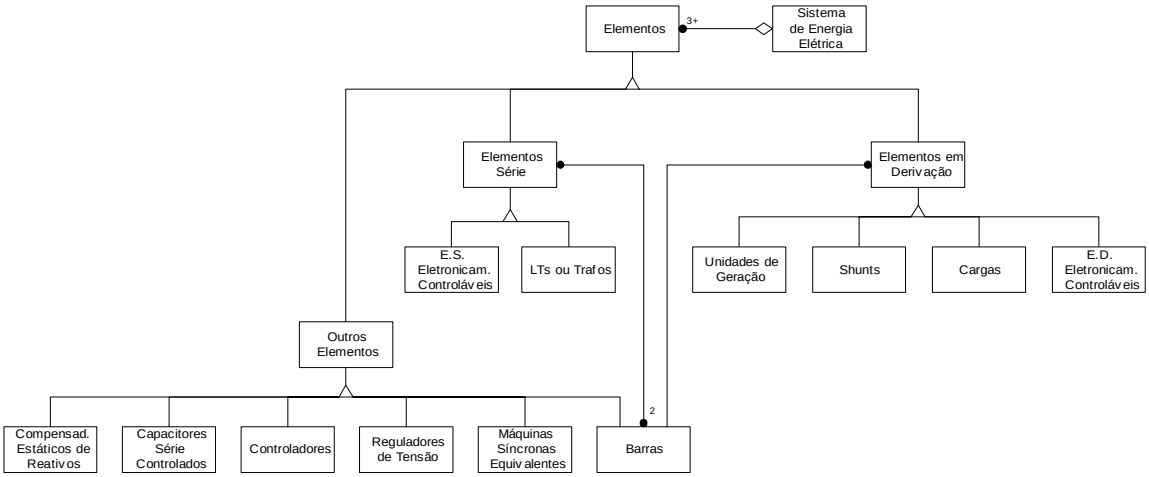

Figura 2: Diagrama de Classes dos Componentes dos SEE

por agregação: o objeto *Unidade de Geração (UnidGer)*. Este é modelado como um agregado de componentes como máquina síncrona equivalente, regulador de tensão e PSS. Este agregado pode ser expandido, incorporando outros componentes à unidade, tais como turbina, caldeira (usinas térmicas) e reguladores de velocidade, os quais não foram modelados neste momento.

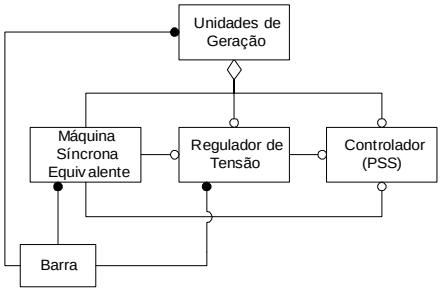

Figura 3: Agregado Unidade de Geração

Nota-se ainda na Figura 3 que os componentes regulador de tensão e PSS são opcionais, pois suas existências dependerão do modelo de máquina utilizado, e do grau de detalhamento desejado para a representação da unidade de geração. O regulador de tensão e o PSS estão conectados à máquina, e também entre si.

Convém observar que todos os diagramas aqui apresentados modelam apenas alguns componentes dos SEE, escolhidos para constituir a base da estrutura hierárquica. As estruturas são completamente expansíveis, no sentido de incorporarem outros componentes do sistema não citados.

4.2 Modelagem de Metodologias

A Figura 4 mostra o diagrama de classes para a representação de metodologias de análise e síntese. Identifica-se uma superclasse *Análises ou Sínteses* (*Analise*), a qual generaliza todas as metodologias, tais como análises da estabilidade transitória (*AETrans*), fluxos de potência (*Flow*), projetos e ajustes de controladores (*PAControl*), análise da estabilidade a pequenos sinais (*AEPSinais*), análises da estabilidade de tensão (*AETensao*), e controles preventivos (*ControlPrev*). Qualquer metodologia de análise ou síntese está associada ao objeto *Sistema de Energia Elétrica*, sendo que este pode estar associado a múltiplas análises ou sínteses.

Na Figura 5 é apresentado o objeto *Fluxo de Potência Linearizado (FlowDC)*, o qual é formado por um objeto *Matriz* (*Matriz*) e um objeto *Sistema Linear (SisLin)*. O objeto Matriz representa a matriz B' e o Sistema Linear representa o sistema de equações $B'.\theta = P$ do modelo de fluxo de potência linearizado.

A Figura 6 apresenta o objeto de análise *Simulação da Dinâmica de SEE com Modelagem Detalhada (SimSP)*. Este objeto é formado por um agregado que contém um objeto *Diagnóstico da Estabilidade (DiagEstab)*, objetos do tipo *Dados de Monitoração (DadMon)*, objetos do tipo *Eventos (Evento)* e um binômio matriz + sistema linear, representando, respectivamente, a matriz da rede elétrica e seu sistema de equações. Os objetos do tipo Eventos modelam as ocorrências a que podem ser submetidos os SEE durante uma simulação, e os do tipo DadMon modelam quais variáveis serão observadas e apresentadas, em forma de gráficos no tempo, ao

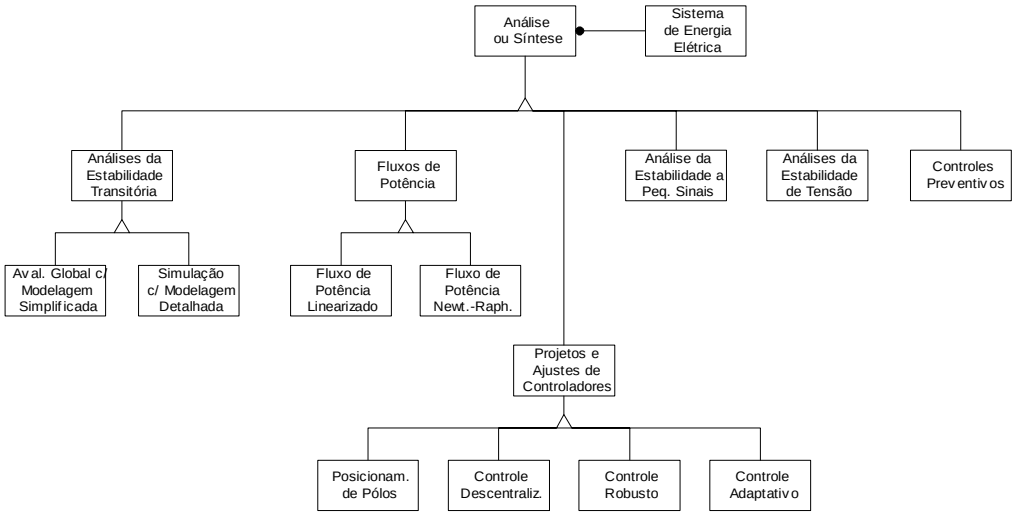

Figura 4: Estrutura de Classes para as Análises e Sínteses

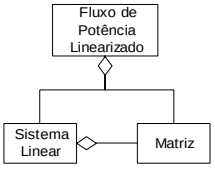

Figura 5: Fluxo de Potência Linearizado

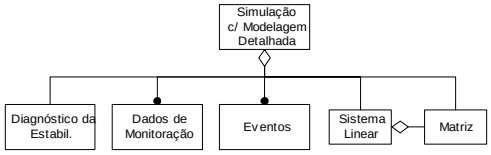

Figura 6: Simulação com Modelagem Detalhada

final de uma simulação.

4.3 Modelagem de Ferramentas para SEE

A seção anterior mostrou as estruturas de classes propostas para a representação das hierarquias de metodologias de análise e síntese. Nessa seção é apresentada uma estrutura hierárquica para a modelagem de ferramentas integradas de análise e síntese.

As *ferramentas* aqui descritas podem ser consideradas como um *produto final*, ou seja, o objeto de mais alto nível em um sistema computacional. São constituídas por um objeto *SEE*, por diversos objetos de análise e síntese, situados nas estruturas da seção anterior, e também por objetos de apoio ao funcionamento de ferramentas específicas. Esses objetos de apoio podem ser matemáticos, tais como objetos para o tratamento de sistemas lineares; ou computacionais tais como listas encadeadas para armazenamento dinâmico de elementos ou objetos para gerenciamento de telas ou tomadas de tempo de execução.

Os objetos *ferramenta* têm um método do tipo *Execute*, o qual quando ativado é responsável pelo funcionamento completo do sistema computacional. Esse método principal encarrega-se de acionar todos os procedimentos seguintes para a execução do programa, tais como inicialização de telas, leituras de comandos (via teclado ou arquivo de controle automático), etc.

Assim, consegue-se um encapsulamento pleno no software, pois a única variável global do projeto é o objeto que representa a ferramenta, e este possui o controle total sobre o programa.

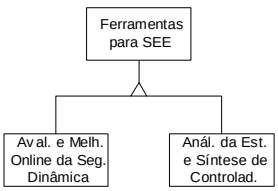

Figura 7: Ferramentas para SEE

A Figura 7 apresenta uma estrutura típica para modelagem de diferentes ferramentas. Nela pode-se notar uma superclasse Ferramentas para SEE (*Ferramentas*), a qual generaliza qualquer ferramenta implemen-

tada. Neste exemplo específico aparecem, na estrutura, duas ferramentas: Avaliação e Melhoria *On-line* da Segurança Dinâmica (*AMOSegDin*) e Análise da Estabilidade e Síntese de Controladores (*AESintControl*).

5 METODOLOGIAS IMPLEMENTADAS

Duas metodologias foram implementadas para validação inicial das estruturas propostas: um Fluxo de Potência Linearizado e uma Simulação da Dinâmica de SEE com Modelagem Detalhada. O objeto para o cálculo do fluxo de potência linearizado é representado pela classe *FlowDC*. Um modelo funcional para esta classe é apresentado na Figura 8, onde podem ser observadas algumas tarefas que a classe realiza.

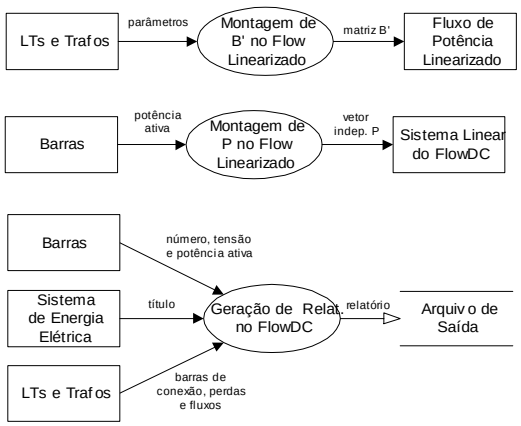

Figura 8: Modelo Funcional para a classe FlowDC

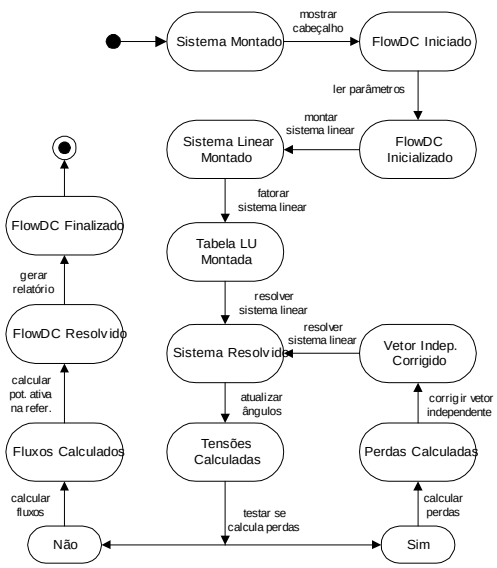

Figura 9: Diagrama de Estados para a Execução do FlowDC

A Figura 9 apresenta um diagrama de estados para o método de execução da classe *FlowDC*. Este é o principal método da classe, e é responsável pela ativação de todos os outros métodos da classe *FlowDC*.

Como pode ser observado no diagrama, a execução do fluxo de potência linearizado inicia com a leitura dos seus parâmetros (arquivo *.fdc*). Após essa etapa, monta o sistema linear, com base nos parâmetros dos elementos que compõe os SEE, resolve o sistema linear, atualiza os ângulos de tensão nos objetos *Barra* e verifica se será realizado cálculo para aproximação das perdas. Em caso positivo, calcula de forma aproximada as perdas ativas nas linhas, corrige o vetor independente P, e resolve novamente o sistema linear. Em caso negativo, calcula (de forma aproximada) os fluxos nas linhas e a potência ativa gerada na barra de referência, gera um relatório de saída com as tensões em cada barra e os fluxos e perdas nas linhas de transmissão, e a execução é encerrada. O relatório de saída é gravado em um arquivo cujo nome é definido através do arquivo de parâmetros do objeto de análise.

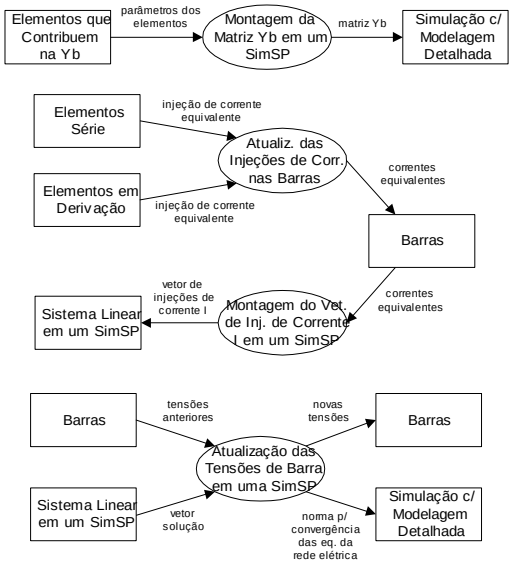

Figura 10: Modelo Funcional para a Classe SimSP

A classe que modela uma simulação da dinâmica com modelagem detalhada é chamada *SimSP*. Na Figura 10 é apresentado um modelo funcional para a classe *SimSP*, onde são mostradas algumas tarefas que este objeto executa, e o fluxo de dados envolvido em cada tarefa.

Na Figura 11 é apresentado o diagrama de estados para a execução da simulação, mostrando a seqüência de operações que é seguida durante a execução.

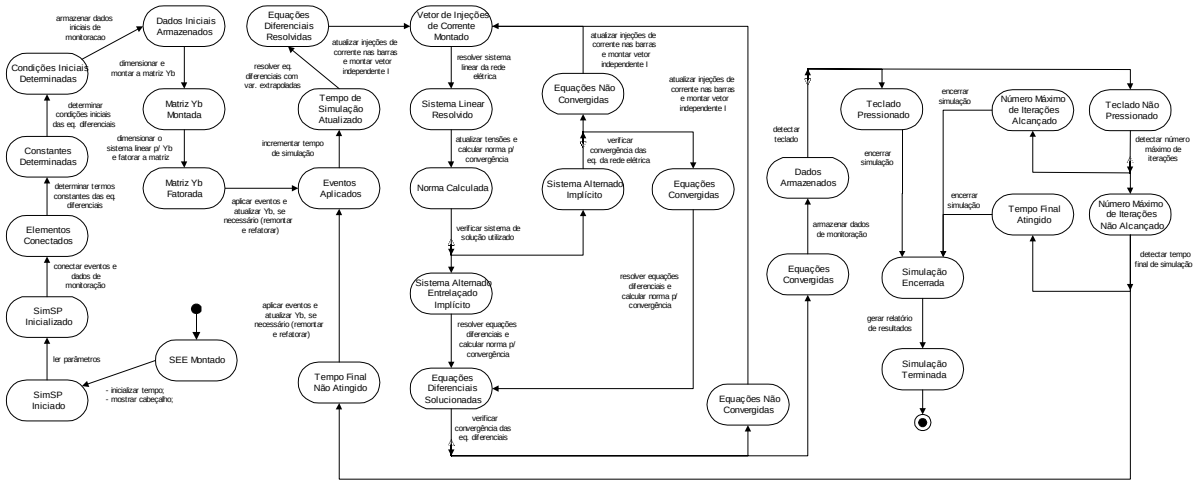

Figura 11: Diagrama de Estados para a Execução do SimSP

6 CONCLUSÕES

Neste trabalho foi apresentada uma proposta de modelagem para SEE utilizando-se MOO. Aspectos gerais sobre a aplicação de MOO na modelagem de SEE foram apresentados, juntamente com as estruturas hierárquicas desenvolvidas.

Duas grandes estruturas foram mostradas: uma representativa dos componentes físicos dos SEE, e outra modelando as metodologias de análise e síntese aplicáveis aos sistemas elétricos, e também as estruturas de ferramentas integradas de análise e síntese.

Dois módulos, representando metodologias de análise para SEE (Fluxo de Potência Linearizado e Simulação da Dinâmica de SEE com Modelagem Detalhada), foram implementados e seus diagramas funcionais e de estados apresentados.

REFERÊNCIAS

Booch, G. (1994). *Object-Oriented Analysis and Design*: With Applications. 2. ed. Addison-Wesley, Reading, Massachusetts.

Coleman, D.; Arnold, P.; Bodoff, S. et al. (1996). *Desenvolvimento Orientado a Objetos* : O Método Fusion. Campus, Rio de Janeiro, RJ.

Eriksson, H. E.; Penker, M. (1998). *UML Toolkit*. Wiley, New York.

Foley, M.; Bose, A. (1995). Object-Oriented On-Line Network Analysis. *IEEE Transactions on Power Systems*, Vol. 10, No. 1, pp. 125-132.

Fuerte-Esquivel, C. R.; Acha, E.; Tan, S. G. et al. (1998). Efficient Object Oriented Power Systems Software for the Analysis of Large-Scale Networks Containing FACTS-Controlled Branches. *IEEE Transactions on Power Systems*, Vol. 13, No. 2, pp. 464-472.

Hakavik, B.; Holen, A. T. (1994). Power System Modelling and Sparse Matrix Operations Using Object-Oriented Programming. *IEEE Transactions on Power Systems*, Vol. 9, No. 2, pp. 1045-1051.

Handschin, E.; Heine, M.; König D. et al. (1998). Object-Oriented Software Engineering for Transmission Planning in Open Access Schemes. *IEEE Transactions on Power Systems*, Vol. 13, No. 1, pp. 94-100.

Manzoni, A.; Silva, A. S.; Decker, I. C. (1999). Power Systems Dynamics Simulation Using Object-Oriented Programming. *IEEE Transactions on Power Systems*, Vol. 14, No. 1, pp. 249-255.

Neyer, A. F.; Wu, F. F.; Imhof, K. (1990). Object-Oriented Programming for Flexible Software: Example of a Load Flow. *IEEE Transactions on Power Systems*, Vol. 5, No. 3, pp. 689-696.

Rumbaugh, J.; Blaha, M.; Premerlani, W. et al. (1994). *Modelagem e Projetos Baseados em Objetos*. Campus, Rio de Janeiro, RJ.

Stroustrup, B. (1997). *The C++ Programming Language*. 3. ed. Addison-Wesley, Reading, Massachusetts.

Zhou, E. Z. (1996). Object-Oriented Programming, C++ and Power System Simulation. *IEEE Transactions on Power Systems*, Vol. 11, No. 1, pp. 206-215.

Zhu, J.; Jossman, P. (1999). Application of Design Patterns for Object-Oriented Modeling of Power Systems. *IEEE Transactions on Power Systems*, Vol. 14, No. 2, pp. 532-537.

Zhu, J.; Lubkeman, D. L. (1997). Object-Oriented Development of Software Systems for Power System Simulations. *IEEE Transactions on Power Systems*, Vol. 12, No. 2, pp. 1002-1007.